

Neural Library

Generated by Doxygen 1.8.5

Tue Mar 11 2014 23:34:57

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	Namespace Documentation	7
4.1	Package NeuralLibrary	7
4.2	Package NeuralLibrary.Neurons	7
5	Class Documentation	9
5.1	NeuralLibrary.Connection Class Reference	9
5.1.1	Detailed Description	10
5.1.2	Constructor & Destructor Documentation	10
5.1.2.1	Connection	10
5.1.2.2	Connection	10
5.1.3	Member Function Documentation	10
5.1.3.1	FeedForward	10
5.1.3.2	NudgeWeight	10
5.1.3.3	UpdateWeight	10
5.1.4	Member Data Documentation	10
5.1.4.1	lastDeltaWeight	10
5.1.5	Property Documentation	11
5.1.5.1	AnteriorNeuron	11
5.1.5.2	Gradient	11
5.1.5.3	PosteriorNeuron	11
5.1.5.4	Weight	11

5.2	NeuralLibrary.DataPoint Class Reference	11
5.3	NeuralLibrary.DataSet Class Reference	11
5.3.1	Detailed Description	12
5.3.2	Member Function Documentation	12
5.3.2.1	CalculateErrors	12
5.3.2.2	Load	12
5.3.2.3	Shuffle	12
5.3.3	Member Data Documentation	13
5.3.3.1	r	13
5.4	NeuralLibrary.FeatureMap Class Reference	13
5.5	NeuralLibrary.Neurons.HiddenNeuron Class Reference	13
5.5.1	Detailed Description	13
5.6	NeuralLibrary.Neurons.InputNeuron Class Reference	13
5.6.1	Detailed Description	14
5.6.2	Member Function Documentation	14
5.6.2.1	UpdateOutput	14
5.7	NeuralLibrary.Network Class Reference	14
5.7.1	Constructor & Destructor Documentation	15
5.7.1.1	Network	15
5.7.1.2	Network	15
5.7.2	Member Function Documentation	15
5.7.2.1	BackPropagate	15
5.7.2.2	FeedForward	16
5.7.2.3	GetConnection	16
5.7.2.4	GetWeights	16
5.7.2.5	Load	16
5.7.2.6	NudgeWeights	16
5.7.2.7	Train	17
5.7.2.8	UpdateWeights	18
5.7.3	Property Documentation	18
5.7.3.1	Bias	18
5.7.3.2	GlobalError	18
5.7.3.3	Input	18
5.7.3.4	Output	18
5.8	NeuralLibrary.Neuron Class Reference	18
5.8.1	Detailed Description	19
5.8.2	Member Function Documentation	19

5.8.2.1	Reset	19
5.8.2.2	UpdateError	19
5.8.2.3	UpdateOutput	20
5.8.3	Property Documentation	20
5.8.3.1	Net	20
5.9	NeuralLibrary.Neurons.OutputNeuron Class Reference	20
5.9.1	Detailed Description	20
5.9.2	Member Function Documentation	21
5.9.2.1	UpdateError	21
5.10	NeuralLibrary.Sigmoid Class Reference	21
5.10.1	Detailed Description	21
5.10.2	Constructor & Destructor Documentation	22
5.10.2.1	Sigmoid	22
5.10.3	Member Data Documentation	23
5.10.3.1	HyperbolicStep	23
5.10.3.2	HyperbolicTangent	23
5.10.3.3	Linear	23
5.10.3.4	Logistic	23
5.10.3.5	None	23
5.10.4	Property Documentation	23
5.10.4.1	Derivative	23
5.10.4.2	Function	24
5.11	NeuralLibrary.Trainer Class Reference	24
5.11.1	Detailed Description	24
5.11.2	Member Function Documentation	24
5.11.2.1	Bound	24
5.11.2.2	Train	25
5.11.3	Property Documentation	26
5.11.3.1	ErrorHistory	26

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

NeuralLibrary	7
NeuralLibrary.Neurons	7

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

NeuralLibrary.Connection	9
NeuralLibrary.DataPoint	11
NeuralLibrary.FeatureMap	13
List< DataPoint >	
NeuralLibrary.DataSet	11
NeuralLibrary.Network	14
NeuralLibrary.Neuron	18
NeuralLibrary.Neurons.HiddenNeuron	13
NeuralLibrary.Neurons.InputNeuron	13
NeuralLibrary.Neurons.OutputNeuron	20
NeuralLibrary.Sigmoid	21
NeuralLibrary.Trainer	24

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

NeuralLibrary.Connection	
The connection held between two neurons with a given weight.	9
NeuralLibrary.DataPoint	11
NeuralLibrary.DataSet	
Holds an abstract dataset.	11
NeuralLibrary.FeatureMap	13
NeuralLibrary.Neurons.HiddenNeuron	
Type specific implementation of the neuron class for hidden neurons.	13
NeuralLibrary.Neurons.InputNeuron	
Type specific implementation of the neuron class for input neurons.	13
NeuralLibrary.Network	14
NeuralLibrary.Neuron	
The base unit of the neural network. Contains pertinent information to neural nodes and feedforward propagation thereof.	18
NeuralLibrary.Neurons.OutputNeuron	
Type specific implementation of the neuron class for output neurons.	20
NeuralLibrary.Sigmoid	
Defines some generic sigmoid activation function and its derivative.	21
NeuralLibrary.Trainer	
Trains the a neural network given a DataSet	24

Chapter 4

Namespace Documentation

4.1 Package NeuralLibrary

Namespaces

- package [Neurons](#)

Classes

- class [Connection](#)
The connection held between two neurons with a given weight.
- class [DataPoint](#)
- class [DataSet](#)
Holds an abstract dataset.
- class [FeatureMap](#)
- class **Gaussian**
- class [Network](#)
- class [Neuron](#)
The base unit of the neural network. Contains pertinent information to neural nodes and feedforward propagation thereof.
- class [Sigmoid](#)
Defines some generic sigmoid activation function and its derivative.
- class [Trainer](#)
Trains the a neural network given a [DataSet](#)

4.2 Package NeuralLibrary.Neurons

Classes

- class [HiddenNeuron](#)
Type specific implementation of the neuron class for hidden neurons.
- class [InputNeuron](#)
Type specific implementation of the neuron class for input neurons.
- class [OutputNeuron](#)
Type specific implementation of the neuron class for output neurons.

Chapter 5

Class Documentation

5.1 NeuralLibrary.Connection Class Reference

The connection held between two neurons with a given weight.

Public Member Functions

- [Connection](#) (double weightInitial, [Neuron](#) anteriorNeuron, [Neuron](#) posteriorNeuron)
Initializes the connection.
- [Connection](#) ([Neuron](#) anteriorNeuron, [Neuron](#) posteriorNeuron)
Initializes the connection with a random weight between -1 and 1
- void [NudgeWeight](#) ()
Nudges the weights.
- void [FeedForward](#) ()
Feeds the product of output from the anterior neuron and the weight of the connection forward to the anterior neuron.
- virtual void [UpdateWeight](#) (double learningRate, double momentum)
*Updates the weight of the connection using the weight update rule. $dW = ERROR_posterior * OUTPUT_anterior$*

Protected Attributes

- double [lastDeltaWeight](#) = 0
The last delta weight (used for momentum)
- double **lastGradient** = 0
- double **acceleration** = 0

Properties

- [Neuron AnteriorNeuron](#) [get, set]
The anterior neuron within the connection.
- [Neuron PosteriorNeuron](#) [get, set]
The posterior neuron within the connection.
- double [Gradient](#) [get]
Gets the gradient of the connection,

- double **Weight** [get, set]
The weight associated with a connection.

5.1.1 Detailed Description

The connection held between two neurons with a given weight.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 NeuralLibrary.Connection.Connection (double *weightInitial*, Neuron *anteriorNeuron*, Neuron *posteriorNeuron*) [inline]

Initializes the connection.

Parameters

<i>weightInitial</i>	The initial weight of the connection. Bound from -1, 1
<i>anteriorNeuron</i>	The neuron on the anterior side of the connection.
<i>posteriorNeuron</i>	The neuron on the posterior side of the connection.

5.1.2.2 NeuralLibrary.Connection.Connection (Neuron *anteriorNeuron*, Neuron *posteriorNeuron*) [inline]

Initializes the connection with a random weight between -1 and 1

Parameters

<i>anteriorNeuron</i>	The neuron on the anterior side of the connection.
<i>posteriorNeuron</i>	The neuron on the posterior side of the connection.

5.1.3 Member Function Documentation

5.1.3.1 void NeuralLibrary.Connection.FeedForward () [inline]

Feeds the product of output from the anterior neuron and the weight of the connection forward to the anterior neuron.

5.1.3.2 void NeuralLibrary.Connection.NudgeWeight () [inline]

Nudges the weights.

5.1.3.3 virtual void NeuralLibrary.Connection.UpdateWeight (double *learningRate*, double *momentum*) [inline], [virtual]

Updates the weight of the connection using the weight update rule. $dW = \text{ERROR_posterior} * \text{OUTPUT_anterior}$

5.1.4 Member Data Documentation

5.1.4.1 double NeuralLibrary.Connection.lastDeltaWeight = 0 [protected]

The last delta weight (used for momentum)

5.1.5 Property Documentation

5.1.5.1 Neuron NeuralLibrary.Connection.AnteriorNeuron [get], [set]

The anterior neuron within the connection.

5.1.5.2 double NeuralLibrary.Connection.Gradient [get]

Gets the gradient of the connection,

5.1.5.3 Neuron NeuralLibrary.Connection.PosteriorNeuron [get], [set]

The posterior neuron within the connection.

5.1.5.4 double NeuralLibrary.Connection.Weight [get], [set]

The weight associated with a connection.

The documentation for this class was generated from the following file:

- Connection.cs

5.2 NeuralLibrary.DataPoint Class Reference

Public Member Functions

- **DataPoint** (double[] input, double[] desired)
- override string **ToString** ()

Properties

- double[] **Input** [get]
- double[] **Desired** [get]

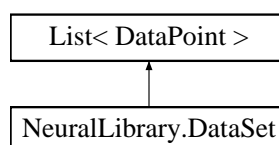
The documentation for this class was generated from the following file:

- DataPoint.cs

5.3 NeuralLibrary.DataSet Class Reference

Holds an abstract dataset.

Inheritance diagram for NeuralLibrary.DataSet:



Public Member Functions

- abstract void [Load](#) ()
Load dataset into the list.
- [DataSet Shuffle](#) ()
An extension for all ILists and dataset which shuffles the order.
- double[] [CalculateErrors](#) ([Network](#) nn, double step=-1)
Calculates the errors based on the datapoints in a given dataset.
- double **CalculateError** ([Network](#) nn, double step=-1)

Static Public Attributes

- static Random [r](#) = new Random()
Random used for the dataset extension (shuffle)

5.3.1 Detailed Description

Holds an abstract dataset.

5.3.2 Member Function Documentation

5.3.2.1 double [] [NeuralLibrary.DataSet.CalculateErrors](#) ([Network](#) nn, double *step* = -1) [inline]

Calculates the errors based on the datapoints in a given dataset.

Parameters

<i>nn</i>	The network from which to calculate the errors
<i>step</i>	The step size.

Returns

5.3.2.2 abstract void [NeuralLibrary.DataSet.Load](#) () [pure virtual]

Load dataset into the list.

5.3.2.3 [DataSet NeuralLibrary.DataSet.Shuffle](#) () [inline]

An extension for all ILists and dataset which shuffles the order.

Template Parameters

<i>T</i>

Parameters

<i>list</i>	
-------------	--

5.3.3 Member Data Documentation

5.3.3.1 Random NeuralLibrary.DataSet.r = new Random() [static]

Random used for the dataset extension (shuffle)

The documentation for this class was generated from the following file:

- DataSet.cs

5.4 NeuralLibrary.FeatureMap Class Reference

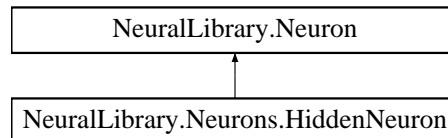
The documentation for this class was generated from the following file:

- FeatureMap.cs

5.5 NeuralLibrary.Neurons.HiddenNeuron Class Reference

Type specific implementation of the neuron class for hidden neurons.

Inheritance diagram for NeuralLibrary.Neurons.HiddenNeuron:



Additional Inherited Members

5.5.1 Detailed Description

Type specific implementation of the neuron class for hidden neurons.

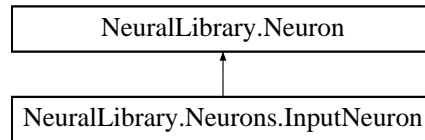
The documentation for this class was generated from the following file:

- Neurons/HiddenNeuron.cs

5.6 NeuralLibrary.Neurons.InputNeuron Class Reference

Type specific implementation of the neuron class for input neurons.

Inheritance diagram for NeuralLibrary.Neurons.InputNeuron:



Public Member Functions

- override void [UpdateOutput](#) ([Sigmoid](#) activation)
Updates the inactivated net for output.

Properties

- override double **Output** [get]

5.6.1 Detailed Description

Type specific implementation of the neuron class for input neurons.

5.6.2 Member Function Documentation

5.6.2.1 override void [NeuralLibrary.Neurons.InputNeuron.UpdateOutput](#) ([Sigmoid activation](#)) [inline],[virtual]

Updates the inactivated net for output.

Reimplemented from [NeuralLibrary.Neuron](#).

The documentation for this class was generated from the following file:

- [Neurons/InputNeuron.cs](#)

5.7 NeuralLibrary.Network Class Reference

Public Member Functions

- [Network](#) (bool RPROP=false, params int[] layers)
Creates a network with default
- [Network](#) (int[] layers, [Sigmoid](#)[] activations, bool RPROP=false)
Constructs a neural network with full control over activations.
- double [Train](#) (double[] input, double[] desired, double learningRate, double momentum)
Trains the neural network using a given input and desired output set.
- void **Save** (string fileName)
- void [NudgeWeights](#) ()
Nudges all of the weights in the network
- void [FeedForward](#) (double[] inputs)
Feeds the network forward achieving an output for a given set of inputs.
- void [BackPropagate](#) (double[] desired)

Propagates the global error backwards through the network using the error backpropagation algorithm

- void [UpdateWeights](#) (double learningRate, double momentum)

Updates all of the weights in the neural network based on neural error.

- [Connection GetConnection](#) (int layer, int anteriorNeuron, int posteriorNeuron)

Gets a connection on a given layer with a given anterior and posterior neuron

- double[] [GetWeights](#) ()

Gets an array of the ewights in the nnetwork.

Static Public Member Functions

- static [Network Load](#) (string fileName, bool RPROP=false)

Loads a neurak network from a file.

Properties

- BiasNeuron [Bias](#) [get, set]

The neural network's bias.

- double[] [Input](#) [get]

Gets the input of the neural network.

- double[] [Output](#) [get]

Gets the output of the neural network.

- double [GlobalError](#) [get, set]

The global error of the network using some squared error.

5.7.1 Constructor & Destructor Documentation

5.7.1.1 NeuralLibrary.Network.Network (bool RPROP = false, params int[] layers) [inline]

Creates a network with default

Parameters

<i>layers</i>	
---------------	--

5.7.1.2 NeuralLibrary.Network.Network (int[] layers, Sigmoid[] activations, bool RPROP = false) [inline]

Constructs a neural network with full control over activations.

Parameters

<i>layers</i>	
<i>activations</i>	

5.7.2 Member Function Documentation

5.7.2.1 void NeuralLibrary.Network.BackPropagate (double[] desired) [inline]

Propagates the global error backwards through the network using the error backpropagation algorithm

Parameters

<i>desired</i>	The set of desired outputs to which the output neurons will be matched
----------------	--

5.7.2.2 `void NeuralLibrary.Network.FeedForward (double[] inputs) [inline]`

Feeds the network forward achieving an output for a given set of inputs.

Parameters

<i>input</i>	The set of inputs to be given to the input neurons.
--------------	---

5.7.2.3 `Connection NeuralLibrary.Network.GetConnection (int layer, int anteriorNeuron, int posteriorNeuron) [inline]`

Gets a connection on a given layer with a given anterior and posterior neuron

Parameters

<i>layer</i>	The layer on which the connection lies
<i>anteriorNeuron</i>	The anterior neuron ID
<i>posteriorNeuron</i>	The posterior neuron ID

Returns

5.7.2.4 `double [] NeuralLibrary.Network.GetWeights () [inline]`

Gets an array of the weights in the network.

Returns

5.7.2.5 `static Network NeuralLibrary.Network.Load (string fileName, bool RPROP = false) [inline],[static]`

Loads a neural network from a file.

Parameters

<i>fileName</i>	The name of the file from which the network will be loaded.
<i>RPROP</i>	Whether or not the network will run the RPROP algorithm

Returns

5.7.2.6 `void NeuralLibrary.Network.NudgeWeights () [inline]`

Nudges all of the weights in the network

5.7.2.7 `double NeuralLibrary.Network.Train (double[] input, double[] desired, double learningRate, double momentum)`
`[inline]`

Trains the neural network using a given input and desired output set.

Parameters

<i>input</i>	The input
<i>desired</i>	The desired output of the neural network
<i>learningRate</i>	The rate at which weights will change
<i>momentum</i>	The momentum with which weight change occurs.

Returns

5.7.2.8 void NeuralLibrary.Network.UpdateWeights (double *learningRate*, double *momentum*) [inline]

Updates all of the weights in the neural network based on neural error.

5.7.3 Property Documentation

5.7.3.1 BiasNeuron NeuralLibrary.Network.Bias [get], [set]

The neural network's bias.

5.7.3.2 double NeuralLibrary.Network.GlobalError [get], [set]

The global error of the network using some squared error.

5.7.3.3 double [] NeuralLibrary.Network.Input [get]

Gets the input of the neural network.

Returns

An array of input values for the neural network.

5.7.3.4 double [] NeuralLibrary.Network.Output [get]

Gets the output of the neural network.

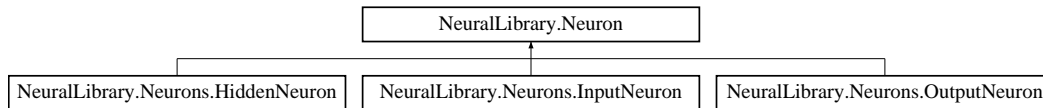
The documentation for this class was generated from the following file:

- Network.cs

5.8 NeuralLibrary.Neuron Class Reference

The base unit of the neural network. Contains pertinent information to neural nodes and feedforward propagation thereof.

Inheritance diagram for NeuralLibrary.Neuron:



Public Member Functions

- void **Reset** ()
Resets the given neuron to its initial state.
- virtual void **UpdateOutput** (**Sigmoid** activation)
Updates the output of the neuron.
- virtual void **UpdateError** (**Sigmoid** activation, double errorCoefficient)
Updates the error of the neuron based on some activation function and some error coefficient (subject to change in Output [Neurons](#)).
- int **GetID** (**Network** network)

Properties

- double **Net** [get, set]
The net input to the sigmoid function of the neuron.
- virtual double **Output** [get, set]
- double **Error** [get, set]

5.8.1 Detailed Description

The base unit of the neural network. Contains pertinent information to neural nodes and feedforward propagation thereof.

5.8.2 Member Function Documentation

5.8.2.1 void NeuralLibrary.Neuron.Reset () [inline]

Resets the given neuron to its initial state.

5.8.2.2 virtual void NeuralLibrary.Neuron.UpdateError (**Sigmoid** activation, double errorCoefficient) [inline], [virtual]

Updates the error of the neuron based on some activation function and some error coefficient (subject to change in Output [Neurons](#)).

Parameters

<i>activation</i>	The activation function with which the error will be calculated.
<i>errorCoefficient</i>	The standard coefficient of error for neurons. $\text{SUM}(\text{for } i \text{ in Posterior } \text{Neurons}) \text{ Error}_i * W_{ij}$. Where j is this neuron.

Returns

The neural error of the neuron.

Reimplemented in [NeuralLibrary.Neurons.OutputNeuron](#).

5.8.2.3 `virtual void NeuralLibrary.Neuron.UpdateOutput (Sigmoid activation)` `[inline],[virtual]`

Updates the output of the neuron.

Parameters

<i>activation</i>	The activation function with which the output is calculated.
-------------------	--

Returns

Reimplemented in [NeuralLibrary.Neurons.InputNeuron](#).

5.8.3 Property Documentation

5.8.3.1 `double NeuralLibrary.Neuron.Net` `[get],[set]`

The net input to the sigmoid function of the neuron.

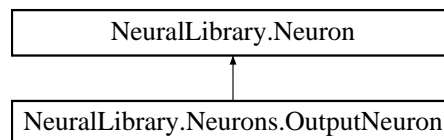
The documentation for this class was generated from the following file:

- Neuron.cs

5.9 NeuralLibrary.Neurons.OutputNeuron Class Reference

Type specific implementation of the neuron class for output neurons.

Inheritance diagram for NeuralLibrary.Neurons.OutputNeuron:

**Public Member Functions**

- override void [UpdateError](#) ([Sigmoid](#) activation, double desired)
Updates the error of the output neuron based on some desired output and a sigmoid activation function.

Additional Inherited Members**5.9.1 Detailed Description**

Type specific implementation of the neuron class for output neurons.

5.9.2 Member Function Documentation

5.9.2.1 `override void NeuralLibrary.Neurons.OutputNeuron.UpdateError (Sigmoid activation, double desired) [inline], [virtual]`

Updates the error of the output neuron based on some desired output and a sigmoid activation function.

Parameters

<i>activation</i>	The activation with which the output will be calculated/
<i>desired</i>	The desired output of this neuron for a given training set.

Reimplemented from [NeuralLibrary.Neuron](#).

The documentation for this class was generated from the following file:

- `Neurons/OutputNeuron.cs`

5.10 NeuralLibrary.Sigmoid Class Reference

Defines some generic sigmoid activation function and its derivative.

Public Member Functions

- [Sigmoid](#) (Func< double, double > function, Func< double, double > derivative)
Defines the sigmoid activation function by a function and its derivative.

Static Public Attributes

- static [Sigmoid HyperbolicTangent](#)
The hyperbolic tangent activation function.
- static [Sigmoid HyperbolicStep](#)
- static [Sigmoid Logistic](#)
- static [Sigmoid Linear](#)
- static [Sigmoid None](#)

Properties

- Func< double, double > [Derivative](#) [get]
The derivative of this sigmoid activation function.
- Func< double, double > [Function](#) [get]
The definition for this sigmoid activation function.

5.10.1 Detailed Description

Defines some generic sigmoid activation function and its derivative.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 `NeuralLibrary.Sigmoid.Sigmoid (Func< double, double > function, Func< double, double > derivative)` `[inline]`

Defines the sigmoid activation function by a function and its derivative.

Parameters

<i>function</i>	
<i>derivative</i>	

5.10.3 Member Data Documentation

5.10.3.1 Sigmoid NeuralLibrary.Sigmoid.HyperbolicStep [static]

Initial value:

```
=
    new Sigmoid(x => Math.Round(HyperbolicTangent.
Function(x)),
    x => Math.Round(HyperbolicTangent.Derivative(x)))
```

5.10.3.2 Sigmoid NeuralLibrary.Sigmoid.HyperbolicTangent [static]

Initial value:

```
=
    new Sigmoid(x => Math.Tanh(x), x => 1 - Math.Pow(Math.Tanh(x), 2))
```

The hyperbolic tangent activation function.

5.10.3.3 Sigmoid NeuralLibrary.Sigmoid.Linear [static]

Initial value:

```
=
    new Sigmoid(x => x, x => 1)
```

5.10.3.4 Sigmoid NeuralLibrary.Sigmoid.Logistic [static]

Initial value:

```
=
    new Sigmoid(x => 1 / (1 + Math.Exp(-x)), x => Math.Exp(x) / Math.Pow(1 + Math.Exp(x), 2)
    )
```

5.10.3.5 Sigmoid NeuralLibrary.Sigmoid.None [static]

Initial value:

```
=
    new Sigmoid(x => 0, x => 0)
```

5.10.4 Property Documentation

5.10.4.1 Func<double, double> NeuralLibrary.Sigmoid.Derivative [get]

The derivative of this sigmoid activation function.

5.10.4.2 Func<double, double> NeuralLibrary.Sigmoid.Function [get]

The definition for this sigmoid activation function.

The documentation for this class was generated from the following file:

- Sigmoid.cs

5.11 NeuralLibrary.Trainer Class Reference

Trains the a neural network given a [DataSet](#)

Public Member Functions

- **Trainer** ([Network](#) network, [DataSet](#) trainingSet)
- **Trainer** ([Network](#) network, [DataSet](#) trainingSet, [DataSet](#) testingSet)
- bool **Train** (int epochs, double minimumError, double learningRate, double momentum, bool nudging=true, double stepPoint=-1, Func< bool > errorCheck=null)

Trains the neural network over a given number of epochs using backpropagation.

- double **Bound** (double val, double min, double max)

Bounds a double to a range.

Properties

- List< double > [ErrorHistory](#) [get, set]

The error history for a given training session (nn);

5.11.1 Detailed Description

Trains the a neural network given a [DataSet](#)

5.11.2 Member Function Documentation

5.11.2.1 double NeuralLibrary.Trainer.Bound (double val, double min, double max) [inline]

Bounds a double to a range.

Parameters

<i>val</i>	
<i>min</i>	
<i>max</i>	

Returns

5.11.2.2 `bool NeuralLibrary.Trainer.Train (int epochs, double minimumError, double learningRate, double momentum, bool nudging = true, double stepPoint = -1, Func< bool > errorCheck = null) [inline]`

Trains the neural network over a given number of epochs using backpropagation.

Parameters

<i>epochs</i>	The number of iterations to which the neural network will train before failing.
<i>minimumError</i>	The minimum error which the network must reach to
<i>learningRate</i>	The learning rate at which the network will begin to learn.
<i>momentum</i>	The momentum at which the network will begin to learn.
<i>nudging</i>	Enables nudging of the neural network during training.

Returns

Whether or not the network was sucessful in learning.

5.11.3 Property Documentation

5.11.3.1 `List<double> NeuralLibrary.Trainer.ErrorHistory` `[get]`, `[set]`

The error history for a given training session (nn);

The documentation for this class was generated from the following file:

- Trainer.cs

Index

AnteriorNeuron
 NeuralLibrary::Connection, 11

BackPropagate
 NeuralLibrary::Network, 15

Bias
 NeuralLibrary::Network, 18

Bound
 NeuralLibrary::Trainer, 24

CalculateErrors
 NeuralLibrary::DataSet, 12

Connection
 NeuralLibrary::Connection, 10

Derivative
 NeuralLibrary::Sigmoid, 23

ErrorHistory
 NeuralLibrary::Trainer, 26

FeedForward
 NeuralLibrary::Connection, 10
 NeuralLibrary::Network, 16

Function
 NeuralLibrary::Sigmoid, 23

GetConnection
 NeuralLibrary::Network, 16

GetWeights
 NeuralLibrary::Network, 16

GlobalError
 NeuralLibrary::Network, 18

Gradient
 NeuralLibrary::Connection, 11

HyperbolicStep
 NeuralLibrary::Sigmoid, 23

HyperbolicTangent
 NeuralLibrary::Sigmoid, 23

Input
 NeuralLibrary::Network, 18

lastDeltaWeight
 NeuralLibrary::Connection, 10

Linear

 NeuralLibrary::Sigmoid, 23

Load
 NeuralLibrary::DataSet, 12
 NeuralLibrary::Network, 16

Logistic
 NeuralLibrary::Sigmoid, 23

Net
 NeuralLibrary::Neuron, 20

Network
 NeuralLibrary::Network, 15

NeuralLibrary, 7

NeuralLibrary.Connection, 9

NeuralLibrary.DataPoint, 11

NeuralLibrary.DataSet, 11

NeuralLibrary.FeatureMap, 13

NeuralLibrary.Network, 14

NeuralLibrary.Neuron, 18

NeuralLibrary.Neurons, 7

NeuralLibrary.Neurons.HiddenNeuron, 13

NeuralLibrary.Neurons.InputNeuron, 13

NeuralLibrary.Neurons.OutputNeuron, 20

NeuralLibrary.Sigmoid, 21

NeuralLibrary.Trainer, 24

NeuralLibrary::Connection
 AnteriorNeuron, 11
 Connection, 10
 FeedForward, 10
 Gradient, 11
 lastDeltaWeight, 10
 NudgeWeight, 10
 PosteriorNeuron, 11
 UpdateWeight, 10
 Weight, 11

NeuralLibrary::DataSet
 CalculateErrors, 12
 Load, 12
 r, 13
 Shuffle, 12

NeuralLibrary::Network
 BackPropagate, 15
 Bias, 18
 FeedForward, 16
 GetConnection, 16
 GetWeights, 16
 GlobalError, 18

- Input, [18](#)
- Load, [16](#)
- Network, [15](#)
- NudgeWeights, [16](#)
- Output, [18](#)
- Train, [16](#)
- UpdateWeights, [18](#)
- NeuralLibrary::Neuron
 - Net, [20](#)
 - Reset, [19](#)
 - UpdateError, [19](#)
 - UpdateOutput, [20](#)
- NeuralLibrary::Neurons::InputNeuron
 - UpdateOutput, [14](#)
- NeuralLibrary::Neurons::OutputNeuron
 - UpdateError, [21](#)
- NeuralLibrary::Sigmoid
 - Derivative, [23](#)
 - Function, [23](#)
 - HyperbolicStep, [23](#)
 - HyperbolicTangent, [23](#)
 - Linear, [23](#)
 - Logistic, [23](#)
 - None, [23](#)
 - Sigmoid, [22](#)
- NeuralLibrary::Trainer
 - Bound, [24](#)
 - ErrorHistory, [26](#)
 - Train, [24](#)
- None
 - NeuralLibrary::Sigmoid, [23](#)
- NudgeWeight
 - NeuralLibrary::Connection, [10](#)
- NudgeWeights
 - NeuralLibrary::Network, [16](#)
- Output
 - NeuralLibrary::Network, [18](#)
- PosteriorNeuron
 - NeuralLibrary::Connection, [11](#)
- r
 - NeuralLibrary::DataSet, [13](#)
- Reset
 - NeuralLibrary::Neuron, [19](#)
- Shuffle
 - NeuralLibrary::DataSet, [12](#)
- Sigmoid
 - NeuralLibrary::Sigmoid, [22](#)
- Train
 - NeuralLibrary::Network, [16](#)
 - NeuralLibrary::Trainer, [24](#)
- UpdateError
 - NeuralLibrary::Neuron, [19](#)
 - NeuralLibrary::Neurons::OutputNeuron, [21](#)
- UpdateOutput
 - NeuralLibrary::Neuron, [20](#)
 - NeuralLibrary::Neurons::InputNeuron, [14](#)
- UpdateWeight
 - NeuralLibrary::Connection, [10](#)
- UpdateWeights
 - NeuralLibrary::Network, [18](#)
- Weight
 - NeuralLibrary::Connection, [11](#)